Washington University Journal of Law & Policy

Volume 30 *Open Source and Proprietary Models of Innovation*

2009

Conceiving Open Systems

Christopher M. Kelty
University of California, Los Angeles

Conceiving Open Systems[†]

Christopher M. Kelty[*]

> The great thing about standards is that there are so many to choose from.[1]

Openness is an unruly concept. While free tends toward ambiguity (free as in speech, or free as in beer?), open tends toward obfuscation. Everyone claims to be open, everyone has something to share, everyone agrees that being open is the obvious thing to do-after all, openness is the other half of "open source"—but for all its obviousness, being "open" is perhaps the most complex component of Free Software. It is never quite clear whether being open is a means or an end. Worse, the opposite of open in this case (specifically, "open systems") is not closed, but "proprietary"—signaling the complicated imbrication of the technical, the legal, and the commercial.

In this Article I tell the story of the contest over the meaning of "open systems" from 1980 to 1993, a contest to create a simultaneously moral and technical infrastructure within the computer industry.[2] The infrastructure in question includes technical

---

† This Article was published previously in substantially similar form as Chapter 5 of CHRISTOPHER M. KELTY, TWO BITS: THE CULTURAL SIGNIFICANCE OF FREE SOFTWARE 14378 (2008), *available at* http://twobits.net/pub/Kelty-TwoBits.pdf. To aid the legal researcher, citations have been reformatted in conformance with THE BLUEBOOK: A UNIFORM SYSTEM OF CITATION (Columbia Law Review Ass'n et al. eds., 18th ed. 2005).

* Christopher M. Kelty is an associate professor at the University of California, Los Angeles. He has a joint appointment in the Center for Society and Genetics and in the department of Information Studies. His research focuses on the cultural significance of information technology, especially in science and engineering. He is the author most recently of TWO BITS: THE CULTURAL SIGNIFICANCE OF FREE SOFTWARE, *supra* note †, as well as numerous articles on open source and free software, including its impact on education, nanotechnology, the life sciences, and issues of peer review and research process in the sciences and in the humanities.

1. T.A. CRITCHLEY & K.C. BATTY, OPEN SYSTEMS: THE REALITY 17 (1993); DON LIBES & SANDY RESSLER, LIFE WITH UNIX: A GUIDE FOR EVERYONE 67 (1989).

2. Moral in this usage signals the "moral and social order" I explored through the

components—the UNIX operating system and the TCP/IP protocols of the Internet as open systems—but it also includes "moral" components, including the demand for structures of fair and open competition, antimonopoly and open markets, and open-standards processes for high-tech networked computers and software in the 1980s.[3] By moral, I mean imaginations of the proper order of collective political and commercial action; referring to much more than simply how individuals should act, moral signifies a vision of how economy and society should be ordered collectively.

The open-systems story is also a story of the blind spot of open systems—in that blind spot is intellectual property. The story reveals a tension between incompatible moral-technical orders: on the one hand, the promise of multiple manufacturers and corporations creating interoperable components and selling them in an open, heterogeneous market; on the other, an intellectual-property system that encouraged jealous guarding and secrecy, and granted monopoly status to source code, designs, and ideas in order to differentiate products and promote competition. The tension proved irresolvable: without shared source code, for instance, interoperable operating systems are impossible. Without interoperable operating systems, internetworking and portable applications are impossible. Without portable applications that can run on any system, open markets are impossible. Without open markets, monopoly power reigns.

Standardization was at the heart of the contest, but by whom and by what means was never resolved. The dream of open systems, pursued in an entirely unregulated industry, resulted in a complicated experiment in novel forms of standardization and cooperation. The

---

concept of social imaginaries in Chapter 1 of KELTY, *supra* note †, at 27–63. Or, in the Scottish Enlightenment sense of Adam Smith, it points to the right organization and relations of exchange among humans. *See generally* ADAM SMITH, THE THEORY OF MORAL SENTIMENTS (2d ed. 1761).

3. There is, of course, a relatively robust discourse of open systems in biology, sociology, systems theory, and cybernetics; however, that meaning of open systems is more or less completely distinct from what openness and open systems came to mean in the computer industry in the period book-ended by the arrivals of the personal computer and the explosion of the Internet (ca. 1980–93). One relevant overlap between these two meanings can be found in the work of Carl Hewitt at the MIT Media Lab and in the interest in "agorics" taken by K. Eric Drexler, Bernardo Huberman, and Mark S. Miller. *See* BERNARDO A. HUBERMAN, THE ECOLOGY OF COMPUTATION (B. A. Huberman ed., 1988).

creation of a "standard" operating system based on UNIX is the story of a failure, a kind of "figuring out" gone haywire, which resulted in huge consortia of computer manufacturers attempting to work together and compete with each other at the same time. Meanwhile, the successful creation of a "standard" networking protocol—known as the Open Systems Interconnection Reference Model ("OSI")—is a story of failure that hides a larger success; OSI was eclipsed in the same period by the rapid and ad hoc adoption of the Transmission Control Protocol/Internet Protocol ("TCP/IP"), which used a radically different standardization process and which succeeded for a number of surprising reasons, allowing the Internet to take the form it did in the 1990s and ultimately exemplifying the moral-technical imagination of a recursive public—and one at the heart of the practices of Free Software.

The conception of openness, which is the central plot of these two stories, has become an essential component of the contemporary practice and power of Free Software. These early battles created a kind of widespread readiness for Free Software in the 1990s, a recognition of Free Software as a removal of open systems' blind spot, as much as an exploitation of its power. The geek ideal of openness and a moral-technical order (the one that made Napster so significant an event) was forged in the era of open systems; without this concrete historical conception of how to maintain openness in technical and moral terms, the recursive public of geeks would be just another hierarchical closed organization—a corporation manqué—and not an independent public serving as a check on the kinds of destructive power that dominated the open-systems contest.

I. HOPELESSLY PLURAL

Big iron, silos, legacy systems, turnkey systems, dinosaurs, mainframes: with the benefit of hindsight, the computer industry of the 1960s to the 1980s appears to be backward and closed, to have literally painted itself into a corner, as an early Intel advertisement suggests.[4] Contemporary observers who show disgust and impatience

4. Intel, Advertisement, *The Difference Between an Open System and Everything Else*, WALL ST. J., May 30, 1984, at 15, *reprinted in* KELTY, *supra* note †, at 146.

with the form that computers took in this era are without fail supporters of open systems and opponents of proprietary systems that "lock in" customers to specific vendors and create artificial demands for support, integration, and management of resources. Open systems (if allowed to flourish) would solve all these problems.

Given the promise of a "general-purpose computer," it should seem ironic at best that open systems needed to be created. But the general-purpose computer never came into being. We do not live in the world of "The Computer," but in a world of computers: myriad, incompatible, specific machines. The design of specialized machines (or "architectures") was, and still is, key to a competitive industry in computers. It required CPUs and components and associated software that could be clearly qualified and marketed as distinct products: the DEC PDP-11 or the IBM 360 or the CDC 6600. On the Fordist model of automobile production, the computer industry's mission was to render desired functions (scientific calculation, bookkeeping, reservations management) in a large box with a button on it (or a very large number of buttons on increasingly smaller boxes). Despite the theoretical possibility, such computers were not designed to do anything, but, rather, to do specific kinds of calculations exceedingly well. They were objects customized to particular markets.

The marketing strategy was therefore extremely stable from about 1955 to about 1980: identify customers with computing needs, build a computer to serve them, provide them with all of the equipment, software, support, or peripherals they need to do the job—and charge a large amount. Organizationally speaking, it was an industry dominated by "IBM and the seven dwarfs": Hewlett-Packard, Honeywell, Control Data, General Electric, NCR, RCA, Univac, and Burroughs, with a few upstarts like DEC in the wings.

By the 1980s, however, a certain inversion had happened. Computers had become smaller and faster; there were more and more of them, and it was becoming increasingly clear to the "big iron" manufacturers that what was most valuable to users was the information they generated, not the machines that did the generating. Such a realization, so the story goes, leads to a demand for interchangeability, interoperability, information sharing, and networking. It also presents the nightmarish problems of conversion between a bewildering, heterogeneous, and rapidly growing array of

hardware, software, protocols, and systems. As one conference paper on the subject of evaluating open systems put it, "At some point a large enterprise will look around and see a huge amount of equipment and software that will not work together. Most importantly, the information stored on these diverse platforms is not being shared, leading to unnecessary duplication and lost profit."[5]

Open systems emerged in the 1980s as the name of the solution to this problem: an approach to the design of systems that, if all participants were to adopt it, would lead to widely interoperable, integrated machines that could send, store, process, and receive the user's information. In marketing and public-relations terms, it would provide "seamless integration."

In theory, open systems was simply a question of standards adoption. For instance, if all the manufacturers of UNIX systems could be convinced to adopt the same basic standard for the operating system, then seamless integration would naturally follow as all the various applications could be written once to run on any variant UNIX system, regardless of which company made it. In reality, such a standard was far from obvious, difficult to create, and even more difficult to enforce. As such, the meaning of open systems was "hopelessly plural," and the term came to mean an incredibly diverse array of things.

"Openness" is precisely the kind of concept that wavers between end and means. Is openness good in itself, or is openness a means to achieve something else—and if so what? Who wants to achieve openness, and for what purpose? Is openness a goal? Or is it a means by which a different goal—say, "interoperability" or "integration"—is achieved? Whose goals are these, and who sets them? Are the goals of corporations different from or at odds with the goals of university researchers or government officials? Are there large central visions to which the activities of all are ultimately subordinate?

Between 1980 and 1993, no person or company or computer industry consortium explicitly set openness as the goal at which

5. Brian William Keves, Open Systems Formal Evaluation Process (Nov. 4, 1993), *in* USENIX SEVENTH SYSTEM ADMINISTRATION CONFERENCE (LISA '93) 87 (1993), *available at* http://www.usenix.org/publications/library/proceedings/lisa93/full_papers/keves.pdf.

organizations, corporations, or programmers should aim, but, by the same token, hardly anyone dissented from the demand for openness. As such, it appears clearly as a kind of cultural imperative, reflecting a longstanding social imagination with roots in liberal democratic notions, versions of a free market and ideals of the free exchange of knowledge, but confronting changed technical conditions that bring the moral ideas of order into relief, and into question.

In the 1980s everyone seemed to want some kind of openness, whether among manufacturers or customers, from General Motors to the armed forces.[6] The debates, both rhetorical and technical, about the meaning of open systems have produced a slough of writings, largely directed at corporate IT managers and CIOs. For instance, Terry A. Critchley and K. C. Batty, the authors of *Open Systems: The Reality*,[7] claim to have collected over a hundred definitions of open systems. The definitions stress different aspects—from interoperability of heterogeneous machines, to compatibility of different applications, to portability of operating systems, to legitimate standards with open-interface definitions—including those that privilege ideologies of a free market, as does Bill Gates's definition: "There's nothing more open than the PC market. . . . [U]sers can choose the latest and greatest software."[8] The range of meanings was huge and oriented along multiple axes: what, to whom, how, and so on. Open systems could mean that source code was open to view or that only the specifications or interfaces were; it could mean "available to certain third parties" or "available to everyone, including competitors"; it could mean self-publishing, well-defined interfaces and application programming interfaces (APIs), or it could mean sticking to standards set by governments and professional societies. To cynics, it simply meant that the marketing department liked the word open and used it a lot.

6. General Motors stirred strong interest in open systems by creating, in 1985, its Manufacturing Automation Protocol ("MAP"), which was built on Unix. At the time, General Motors was the second-largest purchaser of computer equipment after the government. The Department of Defense and the U.S. Air Force also adopted and required POSIX-compliant Unix systems early on.

7. CRITCHLEY & BATTY, *supra* note 1.

8. *Id.* at 11.

One part of the definition, however, was both consistent and extremely important: the opposite of an "open system" was not a "closed system" but a "proprietary system." In industries other than networking and computing the word proprietary will most likely have a positive valence, as in "our exclusive proprietary technology." But in the context of computers and networks such a usage became anathema in the 1980s and 1990s; what customers reportedly wanted was a system that worked nicely with other systems, and that system had to be by definition open since no single company could provide all of the possible needs of a modern business or government agency. And even if it could, it shouldn't be allowed to. For instance:

> In the beginning was the word and the word, was "proprietary." I.B.M. showed the way, purveying machines that existed in splendid isolation. They could not be operated using programs written for any other make of computer; they could not communicate with the machines of competitors.
>
> If your company started out buying computers of various sizes from the International Business Machines Corporation because it was the biggest and the best, you soon found yourself locked as securely to Big Blue as any manacled wretch in a medieval dungeon. When an I.B.M. rival unveiled a technologically advanced product, you could only sigh; it might be years before the new technology showed up in the I.B.M. line.[9]

With the exception of IBM (and to some extent its closest competitors: Hewlett-Packard, Burroughs, and Unisys), computer corporations in the 1980s sought to distance themselves from such "medieval" proprietary solutions (such talk also echoes that of usable pasts of the Protestant Reformation often used by geeks). New firms like Sun and Apollo deliberately berated the IBM model. Bill Joy reportedly called one of IBM's new releases in the 1980s a "grazing

9. Cheryll Aimee Barron, *The Gospel According to Joy*, N.Y. TIMES, Mar. 27, 1988 (Sunday Magazine), at 28.

dinosaur 'with a truck outside pumping its bodily fluids through it.'"[10]

Open systems was never a simple solution though: all that complexity in hardware, software, components, and peripherals could only be solved by pushing hard for standards—even for a single standard. Or, to put it differently, during the 1980s, everyone agreed that open systems was a great idea, but no one agreed on which open systems. As one of the anonymous speakers in *Open Systems: The Reality* puts it, "[i]t took me a long time to understand what (the industry) meant by open vs. proprietary, but I finally figured it out. From the perspective of any one supplier, open meant 'our products.' Proprietary meant 'everyone else's products.'"[11]

For most supporters of open systems, the opposition between open and proprietary had a certain moral force: it indicated that corporations providing the latter were dangerously close to being evil, immoral, perhaps even criminal monopolists. Although there are no doubt arguments for closed systems—security, privacy, robustness, control—the demand for interoperability does not mean that such closure will be sacrificed.[12] Closure was also a choice. That is, open systems was an issue of sovereignty, involving the right, in a moral sense, of a customer to control a technical order hemmed in by firm standards that allowed customers to combine a number of different pieces of hardware and software purchased in an open market and to control the configuration themselves—not enforced openness, but the right to decide oneself on whether and how to be open or closed.

The open-systems idea of moral order conflicts, however, with an idea of moral order represented by intellectual property: the right, encoded in law, to assert ownership over and control particular bits of source code, software, and hardware. The call for and the market in

10. *Dinosaur*, *in* THE ON-LINE HACKER JARGON FILE (Eric Raymond ed., version 4.4.7 2003), http://catb.org/jargon/html/D/dinosaur.html.

11. CRICHTLEY & BATTY, *supra* note 1, at 10.

12. An excellent counterpoint here is PAUL N. EDWARDS, THE CLOSED WORLD: COMPUTERS AND THE POLITICS OF DISCOURSE IN COLD WAR AMERICA (1996). It clearly demonstrates the appeal of a thoroughly and hierarchically controlled system such as the Semi-Automated Ground Environment ("SAGE") of the Department of Defense against the emergence of more "green world" models of openness. *Id.* at 75–111.

open systems were never imagined as being opposed to intellectual property as such, even if the opposition between open and proprietary seemed to indicate a kind of subterranean recognition of the role of intellectual property. The issue was never explicitly broached. Of the hundred definitions in Open Systems, only one definition comes close to including legal issues: "Speaker at Interop '90 (paraphrased and maybe apocryphal): 'If you ask to gain access to a technology and the response you get back is a price list, then that technology is 'open.' If what you get back is a letter from a lawyer, then it's not 'open.'"[13]

Openness here is not equated with freedom to copy and modify, but with the freedom to buy access to any aspect of a system without signing a contract, a nondisclosure agreement, or any other legal document besides a check. The ground rules of competition are unchallenged: the existing system of intellectual property—a system that was expanded and strengthened in this period—was a sine qua non of competition.

Openness understood in this manner means an open market in which it is possible to buy standardized things that are neither obscure nor secret, but can be examined and judged—a "commodity" market, where products have functions, where quality is comparable and forms the basis for vigorous competition. What this notion implies is freedom from monopoly control by corporations over products, a freedom that is nearly impossible to maintain when the entire industry is structured around the monopoly control of intellectual property through trade secret, patent, or copyright. The blind spot hides the contradiction between an industry imagined on the model of manufacturing distinct and tangible products, and the reality of an industry that wavers somewhere between service and product, dealing in intangible intellectual property whose boundaries and identity are in fact defined by how they are exchanged, circulated, and shared, as in the case of the proliferation and differentiation of the UNIX operating system.

There was no disagreement about the necessity of intellectual property in the computer industry of the 1980s, and there was no

13. CRICHTLEY & BATTY, *supra* note 1, at 13.

perceived contradiction in the demands for openness. Indeed, openness could only make sense if it were built on top of a stable system of intellectual property that allowed competitors to maintain clear definitions of the boundaries of their products. But the creation of interoperable components seemed to demand a relaxation of the secrecy and guardedness necessary to "protect" intellectual property. Indeed, for some observers, the problem of openness created the opportunity for the worst kinds of cynical logic, as in this example from Regis McKenna's *Who's Afraid of Big Blue*?

> Users want open environments, so the vendors had better comply. In fact, it is a good idea to support new standards early. That way, you can help control the development of the standards. Moreover, you can take credit for driving the standard. Supporting standards is a way to demonstrate that you're on the side of users.
>
> On the other hand, companies can not compete on the basis of standards alone. Companies that live by standards can die by standards. Other companies, adhering to the same standards, could win on the basis of superior manufacturing technology. If companies do nothing but adhere to standards, then all computers will become commodities, and nobody will be able to make any money.
>
> Thus, companies must keep something proprietary, something to differentiate their products.[14]

By such an account, open systems would be tantamount to economic regression, a state of pure competition on the basis of manufacturing superiority, and not on the basis of the competitive advantage granted by the monopoly of intellectual property, the clear hallmark of a high-tech industry.[15] It was an irresolvable tension

14. REGIS MCKENNA, WHO'S AFRAID OF BIG BLUE? HOW COMPANIES ARE CHALLENGING IBM—AND WINNING 178 (1989). McKenna goes on to suggest that computer companies can differentiate themselves by adding services, better interfaces, or higher reliability—ironically similar to arguments that the Open Source Initiative would make ten years later. *Id.*

15. Richard Stallman, echoing the image of medieval manacled wretches, characterized the blind spot thus:

between the desire for a cooperative, market-based infrastructure and the structure of an intellectual-property system ill-suited to the technical realities within which companies and customers operated—a tension revealing the reorientation of knowledge and power with respect to creation, dissemination, and modification of knowledge.

From the perspective of intellectual property, ideas, designs, and source code are everything—if a company were to release the source code, and allow other vendors to build on it, then what exactly would they be left to sell? Open systems did not mean anything like free, open source, or public domain computing. But the fact that competition required some form of collaboration was obvious as well: standard software and network systems were needed; standard markets were needed; standard norms of innovation within the constraints of standards were needed. In short, the challenge was not just the creation of competitive products but the creation of a standard infrastructure, dealing with the technical questions of availability, modifiability, and reusability of components, and the moral questions of the proper organization of competition and collaboration across diverse domains: engineers, academics, the computer industry, and the industries it computerized. What follows is the story of how UNIX entered the open-systems fray, a story in which the tension between the conceiving of openness and the demands of intellectual property is revealed.

II. OPEN SYSTEMS ONE: OPERATING SYSTEMS

In 1980 UNIX was by all accounts the most obvious choice for a standard operating system for a reason that seemed simple at the outset: it ran on more than one kind of hardware. It had been installed

> Unix does not give the user any more legal freedom than Windows does. What they mean by "open systems" is that you can mix and match components, so you can decide to have, say, a Sun chain on your right leg and some other company's chain on your left leg, and maybe some third company's chain on your right arm, and this is supposed to be better than having to choose to have Sun chains on all of your limbs, or Microsoft chains on all of your limbs. You know, I don't care whose chains are on each limb. What I want is not to be chained by anyone.

Interview by Michael Gross with Richard Stallman, in N.Y., N.Y., and Cambridge, Mass. (1999), *available at* http://www.mgross.com/MoreThgsChng/interviews/stallman5.html.

on DEC machines and IBM machines and Intel processors and Motorola processors—a fact exciting to many professional programmers, university computer scientists, and system administrators, many of whom also considered UNIX to be the best designed of the available operating systems.

There was a problem, however (there always is): UNIX belonged to AT&T, and AT&T had licensed it to multiple manufacturers over the years, in addition to allowing the source code to circulate more or less with abandon throughout the world and to be ported to a wide variety of different machine architectures. Such proliferation, albeit haphazard, was a dream come true: a single, interoperable operating system running on all kinds of hardware. Unfortunately, proliferation would also undo that dream, because it meant that as the markets for workstations and operating systems heated up, the existing versions of UNIX hardened into distinct and incompatible versions with different features and interfaces. By the mid 1980s, there were multiple competing efforts to standardize UNIX, an endeavour that eventually went haywire, resulting in the so-called UNIX wars, in which "gangs" of vendors (some on both sides of the battle) teamed up to promote competing standards. The story of how this happened is instructive, for it is a story that has been reiterated several times in the computer industry.[16]

As a hybrid commercial-academic system, UNIX never entered the market as a single thing. It was licensed in various ways to different people, both academic and commercial, and contained additions and tools and other features that may or may not have originated at (or been returned to) Bell Labs. By the early 1980s, the Berkeley Software Distribution version was in fact competing with the AT&T version, even though BSD was a sublicensee—and it was not the only one. By the late 1970s and early 1980s, a number of corporations had licensed UNIX from AT&T for use on new machines. Microsoft licensed it (and called it Xenix, rather than

16. A similar story can be told about the emergence, in the late 1960s and early 1970s, of manufacturers of "plug-compatible" devices, peripherals that plugged into IBM machines. *See* Shigeru Takahashi, *The Rise and Fall of the Plug-Compatible Mainframes*, IEEE ANNALS HIST. COMPUTING, Jan.–Mar. 2005, at 4, 4–16. Similarly, in the 1990s the story of browser compatibility and the World Wide Web Consortium ("W3C") standards is another recapitulation.

licensing the name UNIX as well) to be installed on Intel-based machines. IBM, Unisys, Amdahl, Sun, DEC, and Hewlett-Packard all followed suit and created their own versions and names: HP-UX, A/UX, AIX, Ultrix, and so on. Given the ground rules of trade secrecy and intellectual property, each of these licensed versions needed to be made legally distinct if they were to compete with each other. Even if UNIX remained conceptually pure in an academic or pedagogical sense, every manufacturer would nonetheless have to tweak, to extend, to optimize in order to differentiate. After all, "[i]f companies do nothing but adhere to standards, then all computers will become commodities, and nobody will be able to make any money."[17]

It was thus unlikely that any of these corporations would contribute the changes they made to UNIX back into a common pool, and certainly not back to AT&T, which subsequent to the 1984 divestiture finally released their own commercial version of UNIX, called UNIX System V. Very quickly, the promising "open" UNIX of the 1970s became a slough of alternative operating systems, each incompatible with the next thanks to the addition of market-differentiating features and hardware-specific tweaks. According to Pamela Gray, "[b]y the mid-1980s, there were more than 100 versions in active use" centered around the three market leaders, AT&T's System V, Microsoft/SCO Xenix, and the BSD.[18] By 1984, the differences in systems had become significant—as in the case of the BSD additions of the TCP/IP protocols, the vi editor, and the Pascal compiler—and created not only differentiation in terms of quality but also incompatibility at both the software and networking levels.

Different systems of course had different user communities, based on who was the customer of whom. Eric Raymond suggests that in the mid-1980s, independent hackers, programmers, and computer scientists largely followed the fortunes of BSD:

---

17. MCKENNA, *supra* note 14, at 178.

18. PAMELA GRAY, OPEN SYSTEMS: A BUSINESS STRATEGY FOR THE 1990S 75, 75 (1991).

> The divide was roughly between longhairs and shorthairs; programmers and technical people tended to line up with Berkeley and BSD, more business-oriented types with AT&T and System V. The longhairs, repeating a theme from Unix's early days ten years before, liked to see themselves as rebels against a corporate empire; one of the small companies put out a poster showing an X-wing-like space fighter marked "BSD" speeding away from a huge AT&T 'death star' logo left broken and in flames.[19]

So even though UNIX had become the standard operating system of choice for time-sharing, multi-user, high-performance computers by the mid-1980s, there was no such thing as UNIX. Competitors in the UNIX market could hardly expect the owner of the system, AT&T, to standardize it and compete with them at the same time, and the rest of the systems were in some legal sense still derivations from the original AT&T system. Indeed, in its licensing pamphlets, AT&T even insisted that UNIX was not a noun, but an adjective, as in "the UNIX system."[20]

The dawning realization that the proliferation of systems was not only spreading UNIX around the world but also spreading it thin and breaking it apart led to a series of increasingly startling and high-profile attempts to "standardize" UNIX. Given that the three major branches (BSD, which would become the industry darling as Sun's Solaris operating system; Microsoft, and later SCO Xenix; and AT&T's System V) all emerged from the same AT&T and Berkeley work done largely by Thompson, Ritchie, and Joy, one would think that standardization would be a snap. It was anything but.

III. FIGURING OUT GOES HAYWIRE

Figuring out the moral and technical order of open systems went haywire around 1986-88, when there were no fewer than four

19. ERIC S. RAYMOND, THE ART OF UNIX PROGRAMMING 35–42 (2004), *available at* http://www.faqs.org/docs/artu/ch02s01.html#id2880014.

20. LIBES & RESSLER, *supra* note 1, at 22; *see also* Andrew Tanenbaum, *The Unix Marketplace in 1987: Life, the Universe and Everything (June 1987)*, *in* PROC. SUMMER 1987 USENIX CONF., June 1987, at 419–24.

competing international standards, represented by huge consortia of computer manufacturers (many of whom belonged to multiple consortia): POSIX, the X/Open consortium, the Open Software Foundation, and UNIX International. The blind spot of open systems had much to do with this crazy outcome: academics, industry, and government could not find ways to agree on standardization. One goal of standardization was to afford customers choice; another was to allow competition unconstrained by "artificial" means. A standard body of source code was impossible; a standard "interface definition" was open to too much interpretation; government and academic standards were too complex and expensive; no particular corporation's standard could be trusted (because they could not be trusted to reveal it in advance of their own innovations); and worst of all, customers kept buying, and vendors kept shipping, and the world was increasingly filled with diversity, not standardization.

UNIX proliferated quickly because of porting, leading to multiple instances of an operating system with substantially similar source code shared by academics and licensed by AT&T. But it differentiated just as quickly because of forking, as particular features were added to different ports. Some features were reincorporated into the "main" branch—the one Thompson and Ritchie worked on—but the bulk of these mutations spread in a haphazard way, shared through users directly or implemented in newly formed commercial versions. Some features were just that, features, but others could extend the system in ways that might make an application possible on one version, but not on another.

The proliferation and differentiation of UNIX, the operating system, had peculiar effects on the emerging market for UNIX, the product: technical issues entailed design and organizational issues. The original UNIX looked the way it did because of the very peculiar structure of the organization that created and sustained UNIX: Bell Labs and the worldwide community of users and developers. The newly formed competitors, conceiving of UNIX as a product distinct from the original UNIX, adopted it precisely because of its portability and because of the promise of open systems as an alternative to "big iron" mainframes. But as UNIX was funneled into existing corporations with their own design and organizational structures, it started to become incompatible with itself, and the desire for

competition in open systems necessitated efforts at UNIX standardization.

The first step in the standardization of open systems and UNIX was the creation of what was called an "interface definition," a standard that enumerated the minimum set of functions that any version of UNIX should support at the interface level, meaning that any programmer who wrote an application could expect to interact with any version of UNIX on any machine in the same way and get the same response from the machine (regardless of the specific implementation of the operating system or the source code that was used). Interface definitions, and extensions to them, were ideally to be published and freely available.

The interface definition was a standard that emphasized portability, not at the source-code or operating-system level, but at the application level, allowing applications built on any version of UNIX to be installed and run on any other. The push for such a standard came first from a UNIX user group founded in 1980 by Bob Marsh and called, after the convention of file hierarchies in the UNIX interface, "/usr/group" (later renamed Uniforum). The 1984 /usr/group standard defined a set of system calls, which, however, "was immediately ignored and, for all practical purposes, useless."[21] It seemed the field was changing too fast and UNIX proliferating and innovating too widely for such a standard to work.

The /usr/group standard nevertheless provided a starting point for more traditional standards organizations—the Institute of Electrical and Electronics Engineers ("IEEE") and the American National Standards Institute ("ANSI")—to take on the task. Both institutions took the /usr/group standard as a basis for what would be called IEEE P1003 Portable Operating System Interface for Computer Environments ("POSIX"). Over the next three years, from 1984 to 1987, POSIX would work diligently at providing a standard interface definition for UNIX.

Alongside this development, the AT&T version of UNIX became the basis for a different standard, the System V Interface Definition ("SVID"), which attempted to standardize a set of functions similar

21. LIBES & RESSLER, *supra* note 1, at 67.

but not identical to the /usr/group and POSIX standards. Thus emerged two competing definitions for a standard interface to a system that was rapidly proliferating into hundreds of tiny operating-system fiefdoms.[22] The danger of AT&T setting the standard was not lost on any of the competing manufacturers. Even if they created a thoroughly open standard-interface definition, AT&T's version of UNIX would be the first to implement it, and they would continually have privileged knowledge of any changes: if they sought to change the implementation, they could change the standard; if they received demands that the standard be changed, they could change their implementation before releasing the new standard.

In response to this threat, a third entrant into the standards race emerged: X/Open, which comprised a variety of European computer manufacturers (including AT&T!) and sought to develop a standard that encompassed both SVID and POSIX. The X/Open initiative grew out of European concern about the dominance of IBM and originally included Bull, Ericsson, ICL, Nixdorf, Olivetti, Philips, and Siemens. In keeping with a certain 1980s taste for the integration of European economic activity vis-à-vis the United States and Japan, these manufacturers banded together both to distribute a unified UNIX operating system in Europe (based initially on the BSD and Sun versions of UNIX) and to attempt to standardize it at the same time.

X/Open represented a subtle transformation of standardization efforts and of the organizational definition of open systems. While the /usr/group standard was developed by individuals who used UNIX, and the POSIX standard by an acknowledged professional society (IEEE), the X/Open group was a collective of computer corporations that had banded together to fund an independent entity to help further the cause of a standard UNIX. This paradoxical situation—of a need to share a standard among all the competitors and the need to keep the details of that standardized product secret to maintain an advantage—was one that many manufacturers, especially

22. A case might be made that a third definition, the ANSI standard for the C programming language, also covered similar ground, which of course it would have had to in order to allow applications written on one operating system to be compiled and run on another. *See* GRAY, *supra* note 18, at 55–58; LIBES & RESSLER, *supra* note 1, at 70–75.

the Europeans with their long experience of IBM's monopoly, understood as mutually destructive. Hence, the solution was to engage in a kind of organizational innovation, to create a new form of metacorporate structure that could strategically position itself as at least temporarily interested in collaboration with other firms, rather than in competition. Thus did stories and promises of open systems wend their way from the details of technical design to those of organizational design to the moral order of competition and collaboration, power and strategy. "Standards" became products that corporations sought to "sell" to their own industry through the intermediary of the consortium.

In 1985 and 1986 the disarrayed state of UNIX was also frustrating to the major U.S. manufacturers, especially to Sun Microsystems, which had been founded on the creation of a market for UNIX-based "workstations," high-powered networked computers that could compete with mainframes and personal computers at the same time. Founded by Bill Joy, Vinod Khosla, and Andreas Bechtolsheim, Sun had very quickly become an extraordinarily successful computer company. The business pages and magazines were keen to understand whether workstations were viable competitors to PCs, in particular to those of IBM and Microsoft, and the de facto standard DOS operating system, for which a variety of extremely successful business-, personal-, and home-computer applications were written.

Sun seized on the anxiety around open systems, as is evident in the ad it ran during the summer of 1987.[23] The ad plays subtly on two anxieties: the first is directed at the consumer and suggests that only with Sun can one actually achieve interoperability among all of one business' computers, much less across a network or industry; the second is more subtle and plays to fears within the computer industry itself, the anxiety that Sun might merge with one of the big corporations, AT&T or Unisys, and corner the market in open systems by producing the de facto standard.[24]

23. Sun Microsystems, Advertisement, *Announcing the Biggest Merger in the Computer Business*, WALL ST. J., July 9, 1987, at 11, *reprinted in* KELTY, *supra* note †, at 159.

24. *Id.*, *reprinted in* KELTY, *supra* note †, at 159.

In fact, in October 1987 Sun announced that it had made a deal with AT&T. AT&T would distribute a workstation based on Sun's SPARC line of workstations and would acquire 20% of Sun.[25] As part of this announcement, Sun and AT&T made clear that they intended to merge two of the dominant versions of UNIX on the market: AT&T's System V and the BSD-derived Solaris. This move clearly frightened the rest of the manufacturers interested in UNIX and open systems, as it suggested a kind of super-power alignment that would restructure (and potentially dominate) the market. A 1988 article in the *New York Times* quotes an industry analyst who characterizes the merger as "a matter of concern at the highest levels of every major computer company in the United States, and possibly the world," and it suggests that competing manufacturers "also fear that A.T.&T. will gradually make Unix a proprietary product, usable only on A.T.&T. or Sun machines."[26] The industry anxiety was great enough that in March Unisys (a computer manufacturer, formerly Burroughs-Sperry) announced that it would work with AT&T and Sun to bring UNIX to its mainframes and to make its business applications run on UNIX. Such a move was tantamount to Unisys admitting that there would be no future in proprietary high-end computing—the business on which it had hitherto built its reputation—unless it could be part of the consortium that could own the standard.[27]

In response to this perceived collusion a group of U.S. and European companies banded together to form another rival organization—one that partially overlapped with X/Open but now included IBM—this one called the Open Software Foundation. A nonprofit corporation, the foundation included IBM, Digital Equipment, Hewlett-Packard, Bull, Nixdorf, Siemens, and Apollo Computer (Sun's most direct competitor in the workstation market). Their goal was explicitly to create a "competing standard" for UNIX that would be available on the hardware they manufactured (and

25. *A.T.&T. Deal with Sun Seen*, N.Y. TIMES, Oct. 19, 1987, at D8.

26. Thomas C. Hayes, *A.T.&T.'s Unix Is a Hit at Last, and Other Companies Are Wary*, N.Y. TIMES, Feb. 24, 1988, at D8.

27. Barnaby J. Feder, *Unisys Obtains Pacts for Unix Capabilities*, N.Y. TIMES, Mar. 10, 1988, at D4.

based, according to some newspaper reports, on IBM's AIX, which was to be called OSF/1). AT&T appeared at first to support the foundation, suggesting that if the Open Software Foundation could come up with a standard, then AT&T would make System V compatible with it. Thus, 1988 was the summer of open love. Every major computer manufacturer in the world was now part of some consortium or another, and some were part of two—each promoting a separate standard.

Of all the corporations, Sun did the most to brand itself as the originator of the open-systems concept. They made very broad claims for the success of open-systems standardization, as for instance in an ad from August 1988, which stated in part:

> But what's more, those sales confirm a broad acceptance of the whole idea behind Sun.
>
> The Open Systems idea. Systems based on standards so universally accepted that they allow combinations of hardware and software from literally thousands of independent vendors. . . . So for the first time, you're no longer locked into the company who made your computers. Even if it's us.[28]

The ad goes on to suggest that "in a free market, the best products win out," even as Sun played both sides of every standardization battle, cooperating with both AT&T and with the Open Software Foundation.[29] But by October of that year, it was clear to Sun that the idea hadn't really become "so universal" just yet. In that month AT&T and Sun banded together with seventeen other manufacturers and formed a rival consortium: Unix International, a coalition of the willing that would back the AT&T UNIX System V version as the one true open standard. In a full-page advertisement from Halloween of 1988, run simultaneously in the *New York Times*, the *Washington Post*, and the *Wall Street Journal*, the rhetoric of achieved success remained, but now instead of "the Open Systems idea," it was "your demand for UNIX System V-based solutions that ushered in the era

28. Sun Microsystems, Advertisement, *It Pays to Be Open*, WALL ST. J., Aug. 2, 1988, at D3, *reprinted in* KELTY, *supra* note †, at 161.

29. *Id.*, *reprinted in* KELTY, *supra* note †, at 161.

of open architecture."[30] Instead of a standard for all open systems, it was a war of all against all, a war to assure customers that they had made, not the right choice of hardware or software, but the right choice of standard.

The proliferation of standards and standards consortia is often referred to as the UNIX wars of the late 1980s, but the creation of such consortia did not indicate clearly drawn lines. Another metaphor that seems to have been very popular in the press at the time was that of "gang" warfare (no doubt helped along by the creation of another industry consortia informally called the Gang of Nine, which were involved in a dispute over whether MicroChannel or EISA buses should be installed in PCs). The idea of a number of companies forming gangs to fight with each other, Bloods-and-Crips style—or perhaps more Jets-and-Sharks style, minus the singing—was no doubt an appealing metaphor at the height of Los Angeles's very real and high-profile gang warfare. But as one article in the *New York Times* pointed out, these were strange gangs:

> Since "openness" and "cooperation" are the buzzwords behind these alliances, the gang often asks its enemy to join. Often the enemy does so, either so that it will not seem to be opposed to openness or to keep tabs on the group. I.B.M. was invited to join the corporation for Open Systems, even though the clear if unstated motive of the group was to dilute I.B.M.'s influence in the market. A.T.&T. negotiated to join the Open Software Foundation, but the talks collapsed recently.
>
> Some companies find it completely consistent to be members of rival gangs. . . . About 10 companies are members of both the Open Software Foundation and its archrival, Unix International.[31]

---

30. Unix Int'l, Advertisement, *If You Believe Unix System V Is the Open Standard, You're Not Alone*, N.Y. TIMES, Oct. 31, 1988, at D3, *reprinted in* KELTY, *supra* note †, at 163.

31. Andrew Pollack, *Computer 'Gangs' Stake Out Turf*, N.Y. TIMES, Dec. 13, 1988, at D1; *see also* Evelyn Richards, *Computer Industry Slips into Disarray as Squabbling Grows*, WASH. POST, Dec. 11, 1988, at K1; Brit Hume, *IBM, Once the Bully on the Block, Faces a Tough New PC Gang*, WASH. POST, Oct. 3, 1988, at E24.

The proliferation of these consortia can be understood in various ways. One could argue that they emerged at a time-during the Reagan administration—when antitrust policing had diminished to the point where computer corporations did not see such collusion as a risky activity vis-à-vis antitrust policing. One could also argue that these consortia represented a recognition that the focus on hardware control (the meaning of proprietary) had been replaced with a focus on the control of the "open standard" by one or several manufacturers, that is, that competition was no longer based on superior products, but on "owning the standard." It is significant that the industry consortia quickly overwhelmed national efforts, such as the IEEE POSIX standard, in the media—an indication that no one was looking to government or nonprofits, or to university professional societies, to settle the dispute by declaring a standard, but rather to industry itself to hammer out a standard, de facto or otherwise. Yet another way to understand the emergence of these consortia is as a kind of mutual policing of the market, a kind of paranoid strategy of showing each other just enough to make sure that no one would leapfrog ahead and kill the existing, fragile competition.

What this proliferation of UNIX standards and consortia most clearly represents, however, is the blind spot of open systems: the difficulty of having collaboration and competition at the same time in the context of intellectual-property rules that incompletely capture the specific and unusual characteristics of software. For participants in this market, the structure of intellectual property was unassailable—without it, most participants assumed, innovation would cease and incentives disappear. Despite the fact that secrecy haunted the industry, its customers sought both openness and compatibility. These conflicting demands proved irresolvable.

IV. DENOUEMENT

Ironically, the UNIX wars ended not with the emergence of a winner, but with the reassertion of proprietary computing: Microsoft Windows and Windows NT. Rather than open systems emerging victorious, ushering in the era of seamless integration of diverse components, the reverse occurred: Microsoft managed to grab a huge share of computer markets, both desktop and high-performance, by

leveraging its brand, the ubiquity of DOS, and application-software developers' dependence on the "Wintel" monster (Windows plus Intel chips). Microsoft triumphed, largely for the same reasons the open-systems dream failed: the legal structure of intellectual property favored a strong corporate monopoly on a single, branded product over a weak array of "open" and competing components. There was no large gain to investors, or to corporations, from an industry of nice guys sharing the source code and making the components work together. Microsoft, on the other hand, had decided to do so internal to itself; it did not necessarily need to form consortia or standardize its operating systems, if it could leverage its dominance in the market to spread the operating system far and wide. It was, as standards observers like to say, the triumph of de facto standardization over de jure. It was a return to the manacled wretches of IBM's monopoly—but with a new dungeon master.

The denouement of the UNIX standards story was swift: AT&T sold its UNIX System Labs (including all of the original source and rights) to Novell in 1993, who sold it in turn to SCO two years later. Novell sold (or transferred) the trademark name UNIX™ to the X/Open group, which continued to fight for standardization, including a single universal UNIX specification. In 1996 X/Open and the Open Software Foundation merged to form the Open Group.[32] The Open Group eventually joined forces with IEEE to turn POSIX into a single UNIX specification in 2001. They continue to push the original vision of open systems, though they carefully avoid using the name or concept, referring instead to the trademarked mouthful "Boundaryless Information Flow" and employing an updated and newly inscrutable rhetoric: "Boundaryless Information Flow, a shorthand representation of 'access to integrated information to support business process improvements' represents a desired state of an enterprise's infrastructure and is specific to the business needs of the organization."[33]

32. The Unix System—History and Timeline, http://www.unix.org/what_is_unix/history_timeline.html (last visited Feb. 27, 2009).

33. The Open Group Vision and Mission, http://www.opengroup.org/overview/vision-mission.htm (last visited Feb. 27, 2009).

The Open Group, as well as many other participants in the history of open systems, recognize the emergence of "open source" as a return to the now one true path of boundaryless information flow. Eric Raymond, of course, sees continuity and renewal (not least of which is in his own participation in the Open Source movement) and in his Art of UNIX Programming says, "The Open Source movement is building on this stable foundation and is creating a resurgence of enthusiasm for the UNIX philosophy. In many ways Open Source can be seen as the true delivery of Open Systems that will ensure it continues to go from strength to strength."[34]

This continuity, of course, deliberately disavows the centrality of the legal component, just as Raymond and the Open Source Initiative had in 1998. The distinction between a robust market in UNIX operating systems and a standard UNIX-based infrastructure on which other markets and other activities can take place still remains unclear even to those closest to the money and machines. It does not yet exist, and may well never come to.

The growth of Free Software in the 1980s and 1990s depended on openness as a concept and component that was figured out during the UNIX wars. It was during these wars that the Free Software Foundation (and other groups, in different ways) began to recognize the centrality of the issue of intellectual property to the goal of creating an infrastructure for the successful creation of open systems.[35] The GNU (GNU's Not Unix) project in particular, but also the X Windows system at MIT, the Remote Procedure Call and Network File System ("NFS") systems created by Sun, and tools like sendmail and BIND were each in their own way experiments with alternative licensing arrangements and were circulating widely on a variety of the UNIX versions in the late 1980s. Thus, the experience of open systems, while technically a failure as far as UNIX was concerned, was nonetheless a profound learning experience for an entire generation of engineers, hackers, geeks, and entrepreneurs. Just as the UNIX operating system had a pedagogic life of its own,

34. The Unix System, *supra* note 32.

35. Larry McVoy was an early voice, within Sun, arguing for solving the open-systems problem by turning to Free Software. Larry McVoy, The Sourceware Operating System Proposal (Nov. 9, 1993), http://www.bitmover.com/lm/papers/srcos.html.

inculcating itself into the minds of engineers as the paradigm of an operating system, open systems had much the same effect, realizing an inchoate philosophy of openness, interconnection, compatibility, and interoperability—in short, availability and modifiability—that was in conflict with intellectual-property structures as they existed. To put it in Freudian terms: the neurosis of open systems was not cured, but the structure of its impossibility had become much clearer to everyone. UNIX, the operating system, did not disappear at all—but UNIX, the market, did.

V. OPEN SYSTEMS TWO: NETWORKS

The struggle to standardize UNIX as a platform for open systems was not the only open-systems struggle; alongside the UNIX wars, another "religious war" was raging. The attempt to standardize networks—in particular, protocols for the inter-networking of multiple, diverse, and autonomous networks of computers—was also a key aspect of the open-systems story of the 1980s.[36] The war between the TCP/IP and OSI was also a story of failure and surprising success: the story of a successful standard with international approval (the OSI protocols) eclipsed by the experimental, military-funded TCP/IP, which exemplified an alternative and unusual standards process. The moral-technical orders expressed by OSI and TCP/IP are, like that of UNIX, on the border between government, university, and industry; they represent conflicting social imaginations in which power and legitimacy are organized differently and, as a result, expressed differently in the technology.

OSI and TCP/IP started with different goals: OSI was intended to satisfy everyone, to be the complete and comprehensive model against which all competing implementations would be validated;

36. The distinction between a protocol, an implementation and a standard is important: Protocols are descriptions of the precise terms by which two computers can communicate (i.e., a dictionary and a handbook for communicating). An implementation is the creation of software that uses a protocol (i.e., actually does the communicating); thus two implementations using the same protocol should be able to share data. A standard defines which protocol should be used by which computers, and for what purposes. It may or may not define the protocol, but will set limits on changes to that protocol.

TCP/IP, by contrast, emphasized the easy and robust interconnection of diverse networks. TCP/IP is a protocol developed by bootstrapping between standard and implementation, a mode exemplified by the Requests for Comments system that developed alongside them as part of the Arpanet project. OSI was a "model" or reference standard developed by internationally respected standards organizations.

In the mid-1980s OSI was en route to being adopted internationally, but by 1993 it had been almost completely eclipsed by TCP/IP. The success of TCP/IP is significant for three reasons: (1) availability—TCP/IP was itself available via the network and development was open to anyone, whereas OSI was a bureaucratically confined and expensive standard and participation was confined to state and corporate representatives, organized through ISO in Geneva; (2) modifiability—TCP/IP could be copied from an existing implementation (such as the BSD version of UNIX) and improved, whereas OSI was a complex standard that had few existing implementations available to copy; and (3) serendipity—new uses that took advantage of availability and modifiability sprouted, including the "killer app" that was the World Wide Web, which was built to function on existing TCP/IP—based networks, convincing many manufacturers to implement that protocol instead of, or in addition to, OSI.

The success of TCP/IP over OSI was also significant because of the difference in the standardization processes that it exemplified. The OSI standard (like all official international standards) is conceived and published as an aid to industrial growth: it was imagined according to the ground rules of intellectual property and as an attempt to facilitate the expansion of markets in networking. OSI would be a "vendor-neutral" standard: vendors would create their own, secret implementations that could be validated by OSI and thereby be expected to interoperate with other OSI-validated systems. By stark contrast, the TCP/IP protocols were not published (in any conventional sense), nor were the implementations validated by a legitimate international-standards organization; instead, the protocols are themselves represented by implementations that allow connection to the network itself (where the TCP/IP protocols and implementations are themselves made available). The fact that one can only join the network if one possesses or makes an

implementation of the protocol is generally seen as the ultimate in validation: it works.[37] In this sense, the struggle between TCP/IP and OSI is indicative of a very familiar twentieth-century struggle over the role and extent of government planning and regulation (versus entrepreneurial activity and individual freedom), perhaps best represented by the twin figures of Friedrich Hayek and Maynard Keynes. In this story, it is Hayek's aversion to planning and the subsequent privileging of spontaneous order that eventually triumphs, not Keynes's paternalistic view of the government as a neutral body that absorbs or encourages the swings of the market.

VI. BOOTSTRAPPING NETWORKS

The "religious war" between TCP/IP and OSI occurred in the context of intense competition among computer manufacturers and during a period of vibrant experimentation with computer networks worldwide. As with most developments in computing, IBM was one of the first manufacturers to introduce a networking system for its machines in the early 1970s: the System Network Architecture ("SNA"). DEC followed suit with Digital Network Architecture ("DECnet" or "DNA"), as did Univac with Distributed Communications Architecture ("DCA"), Burroughs with Burroughs Network Architecture ("BNA"), and others. These architectures were, like the proprietary operating systems of the same era, considered closed networks, networks that interconnected a centrally planned and specified number of machines of the same type or made by the same manufacturer. The goal of such networks was to make connections internal to a firm, even if that involved geographically widespread systems (e.g., from branch to headquarters). Networks were also to be products.

The 1970s and 1980s saw extraordinarily vibrant experimentation with academic, military, and commercial networks. Robert Metcalfe had developed Ethernet at Xerox PARC in the mid-1970s, and IBM

37. The advantages of such an unplanned and unpredictable network have come to be identified in hindsight as a design principle. For an excellent analysis of the history of "end to end" or "stupid" networks, see Tarleton Gillespie, *Engineering a Principle: End-to-End in the Design of the Internet*, 36 SOC. STUD. SCI. 427, 441–50 (2006).

later created a similar technology called "token ring." In the 1980s the military discovered that the Arpanet was being used predominantly by computer scientists and not just for military applications, and decided to break it into MILNET and CSNET.[38] Bulletin Board Services, which connected PCs to each other via modems to download files, appeared in the late 1970s. Out of this grew Tom Jennings's very successful experiment called FidoNet.[39] In the 1980s an existing social network of university faculty on the East Coast of the United States started a relatively successful network called BITNET (Because It's There Network) in the mid-1980s.[40] The Unix to Unix Copy Protocol ("uucp"), which initially enabled the Usenet, was developed in the late 1970s and widely used until the mid-1980s to connect UNIX computers together. In 1984 the NSF began a program to fund research in networking and created the first large backbones for NSFNet, successor to the CSNET and Arpanet.[41]

In the 1970s telecommunications companies and spin-off start-ups experimented widely with what were called "videotex" systems, of which the most widely implemented and well known is Minitel in France.[42] Such systems were designed for consumer users and often provided many of the now widespread services available on the Internet in a kind of embryonic form (from comparison shopping for cars, to directory services, to pornography).[43] By the late 1970s, videotex systems were in the process of being standardized by the Commité Consultative de Information, Technologie et Télécommunications ("CCITT") at the International Telecommunications Union ("ITU") in Geneva. These standards

38. William J. Broad, *Global Computer Network Split as Safeguard*, N.Y. TIMES, Oct. 5, 1983, at A13.

39. *See* DVD: BBS: The Documentary (Bovine Ignition Systems, 2005), *available at* http://www.bbsdocumentary.com/.

40. David Alan Grier & Mary Campbell, *A Social History of Bitnet and Listserv 1985–1991*, IEEE ANNALS HIST. COMPUTING, Apr.–June 2000, at 32, 33.

41. *See* MICHAEL HAUBEN & RONDA HAUBEN, NETIZENS: ON THE HISTORY AND IMPACT OF USENET AND THE INTERNET (1997); *see also* Bryan Pfaffenberger, *"A Standing Wave in the Web of Our Communications": Usenet and the Socio-Technical Construction of Cyberspace Values*, *in* FROM USENET TO COWEBS: INTERACTING WITH SOCIAL INFORMATION SPACES 20, 20–43 (Christopher Lueg & Danyel Fisher eds., 2003).

42. SUSANNE K. SCHMIDT & RAYMUND WERLE, COORDINATING TECHNOLOGY: STUDIES IN THE INTERNATIONAL STANDARDIZATION OF TELECOMMUNICATIONS 147–84 (1998).

43. *See, e.g.*, JAMES MARTIN, VIEWDATA AND THE INFORMATION SOCIETY (1982).

efforts would eventually be combined with work of the International Organization for Standardization ("ISO") on OSI, which had originated from work done at Honeywell.[44]

One important feature united almost all of these experiments: the networks of the computer manufacturers were generally piggybacked, or bootstrapped, onto existing telecommunications infrastructures built by state-run or regulated monopoly telecommunications firms. This situation inevitably spelled grief, for telecommunications providers are highly regulated entities, while the computer industry has been almost totally unregulated from its inception. Since an increasingly core part of the computer industry's business involved transporting signals through telecommunications systems without being regulated to do so, the telecommunications industry naturally felt themselves at a disadvantage.[45] Telecommunications companies were not slow to respond to the need for data communications, but their ability to experiment with products and practices outside the scope of telephony and telegraphy was often hindered by concerns about antitrust and monopoly.[46] The unregulated computer industry, by contrast, saw the tentativeness of the telecommunications industry (or national PTTs) as either bureaucratic inertia or desperate attempts to maintain control and power over existing networks—though no computer manufacturer relished the idea of building their own physical network when so many already existed.

TCP/IP and OSI have become emblematic of the split between the worlds of telecommunications and computing; the metaphors of religious wars or of blood feuds and cold wars were common.[47] A

---

44. There is little information on the development of open systems; there is, however, a brief note from William Stallings, author of perhaps the most widely used textbook on networking. William Stallings, The Origins of OSI (1998), http://williamstallings.com/Extras/OSI.html.

45. GERALD W. BROCK, THE SECOND INFORMATION REVOLUTION (2003), is a good introductory source for this conflict, at least in its policy outlines. The Federal Communications Commission issued two decisions (known as "Computer 1" and "Computer 2") that attempted to deal with this conflict by trying to define what counted as voice communication and what as data. *See In re* Regulatory & Policy Problems Presented by the Interdependence of Computer & Commc'n Servs. & Facilities, 28 F.C.C. 2d 291 (1970) (known as Computer 1); *In re* Second Computer Inquiry, 77 F.C.C. 2d 384 (1980) (known as Computer 2), *superseded by regulation as stated in In re* N.J. Bell Telephone Co., 8 F.C.C.R. 5153 (1993).

46. *See* BROCK, *supra* note 45, at 170–85.

47. *See, e.g.*, William J. Drake, *The Internet Religious War*, 17 TELECOMM. POL'Y 643,

particularly arch account from this period is Carl Malamud's *Exploring the Internet: A Technical Travelogue*, which documents Malamud's (physical) visits to Internet sites around the globe, discussions (and beer) with networking researchers on technical details of the networks they have created, and his own typically geeky, occasionally offensive takes on cultural difference.[48] A subtheme of the story is the religious war between Geneva (in particular the ITU) and the Internet: Malamud tells the story of asking the ITU to release its 19,000-page "blue book" of standards on the Internet, to facilitate its adoption and spread.[49]

The resistance of the ITU and Malamud's heroic if quixotic attempts are a parable of the moral-technical imaginaries of openness—and indeed, his story draws specifically on the usable past of Giordano Bruno.[50] The "bruno" project demonstrates the gulf that exists between two models of legitimacy—those of ISO and the ITU—in which standards represent the legal and legitimate consensus of a regulated industry, approved by member nations, paid for and enforced by governments, and implemented and adhered to by corporations.

Opposite ISO is the ad hoc, experimental style of Arpanet and Internet researchers, in which standards are freely available and implementations represent the mode of achieving consensus, rather than the outcome of the consensus. In reality, such a rhetorical opposition is far from absolute: many ISO standards are used on the Internet, and ISO remains a powerful, legitimate standards organization. But the clash of established (telecommunications) and emergent (computer-networking) industries is an important context for understanding the struggle between OSI and TCP/IP.

643–49 (1993).

48. CARL MALAMUD, EXPLORING THE INTERNET: A TECHNICAL TRAVELOGUE (1992); *see also* Michael M. J. Fischer, *Worlding Cyberspace: Toward a Critical Ethnography in Time, Space, and Theory*, *in* CRITICAL ANTHROPOLOGY NOW 245, 245–304 (George E. Marcus ed., 1999).

49. MALAMUD, *supra* note 48, at 5.

50. The usable past of Giordano Bruno is invoked by Malamud to signal the heretical nature of his own commitment to openly publishing standards that ISO was opposed to releasing. Bruno's fate at the hands of the Roman Inquisition hinged in some part on his acceptance of the Copernican cosmology, so he has been, like Galileo, a natural figure for revolutionary claims during the 1990s. *Id*. at 35.

The need for standard networking protocols is unquestioned: interoperability is the bread and butter of a network. Nonetheless, the goals of the OSI and the TCP/IP protocols differed in important ways, with profound implications for the shape of that interoperability. OSI's goals were completeness, control, and comprehensiveness. OSI grew out of the telecommunications industry, which had a long history of confronting the vicissitudes of linking up networks and facilitating communication around the world, a problem that required a strong process of consensus and negotiation among large, powerful, government-run entities, as well as among smaller manufacturers and providers. OSI's feet were firmly planted in the international standardization organizations like OSI and the ITU (an organization as old as telecommunications itself, dating to the 1860s).

Even if they were oft-mocked as slow, bureaucratic, or cumbersome, the processes of ISO and ITU-based in consensus, international agreement, and thorough technical specification—are processes of unquestioned legitimacy. The representatives of nations and corporations who attend ISO and ITU standards discussions, and who design, write, and vote on these standards, are usually not bureaucrats, but engineers and managers directly concerned with the needs of their constituency. The consensus-oriented process means that ISO and ITU standards attempt to satisfy all members' goals, and as such they tend to be very large, complex, and highly specific documents. They are generally sold to corporations and others who need to use them, rather than made freely available, a fact that until recently reflected their legitimacy, rather than lack thereof.

TCP/IP, on the other hand, emerged from very different conditions.[51] These protocols were part of a Department of Defense-funded experimental research project: Arpanet. The initial Arpanet

51. JANET ABBATE, INVENTING THE INTERNET 133–45 (1999); BROCK, *supra* note 45, at 146–51; ALEXANDER GALLOWAY, PROTOCOL: HOW CONTROL EXISTS AFTER DECENTRALIZATION 3–9 (2004); PETER H. SALUS, CASTING THE NET: FROM ARPANET TO INTERNET AND BEYOND (1995). For practitioner histories, see David D. Clark, *The Design Philosophy of the DARPA Internet Protocols*, *in* COMPUTER COMMUNICATIONS: ARCHITECTURES, PROTOCOLS, AND STANDARDS 54, 54–62 (William Stallings ed., 1992); Robert Kahn et al., *The Evolution of the Internet as a Global Information System*, 29 INT'L INFO. & LIBR. REV. 129 (1997).

protocols (the Network Control Protocol, or NCP) were insufficient, and TCP/IP was an experiment in interconnecting two different "packet-switched networks": the ground-line-based Arpanet network and a radio-wave network called Packet Radio.[52] The problem facing the designers was not how to accommodate everyone, but merely how to solve a specific problem: interconnecting two technically diverse networks, each with autonomous administrative boundaries, but forcing neither of them to give up the system or the autonomy.

Until the mid-1980s, the TCP/IP protocols were resolutely research-oriented, and not the object of mainstream commercial interest. Their development reflected a core set of goals shared by researchers and ultimately promoted by the central funding agency, the Department of Defense. The TCP/IP protocols are often referred to as enabling packet-switched networks, but this is only partially correct; the real innovation of this set of protocols was a design for an "inter-network," a system that would interconnect several diverse and autonomous networks (packet-switched or circuit-switched), without requiring them to be transformed, redesigned, or standardized—in short, by requiring only standardization of the intercommunication between networks, not standardization of the network itself. In the first paper describing the protocol Robert Kahn and Vinton Cerf motivated the need for TCP/IP thus:

> Even though many different and complex problems must be solved in the design of an individual packet-switching network, these problems are manifestly compounded when dissimilar networks are interconnected. Issues arise which may have no direct counterpart in an individual network and which strongly influence the way in which Internetwork communication can take place.[53]

The explicit goal of TCP/IP was thus to share computer resources, not necessarily to connect two individuals or firms together, or to create a competitive market in networks or networking software. Sharing between different kinds of networks implied allowing the

52. *See* ABBATE, *supra* note 51, at 114–36; Kahn et al., *supra* note 51, at 134–40.

53. Vinton G. Cerf & Robert E. Kahn, *A Protocol for Packet Network Intercommunication*, 22 IEEE TRANSACTIONS ON COMM. 637, 637 (1974).

different networks to develop autonomously (as their creators and maintainers saw best), but without sacrificing the ability to continue sharing. Years later, David Clark, chief Internet engineer for several years in the 1980s, gave a much more explicit explanation of the goals that led to the TCP/IP protocols. In particular, he suggested that the main overarching goal was not just to share resources but "to develop an effective technique for multiplexed utilization of existing interconnected networks,"[54] and he more explicitly stated the issue of control that faced the designers: "[N]etworks represent administrative boundaries of control, and it was an ambition of this project to come to grips with the problem of integrating a number of separately administrated entities into a common utility."[55] By placing the goal of expandability first, the TCP/IP protocols were designed with a specific kind of simplicity in mind: the test of the protocols' success was simply the ability to connect.

By setting different goals, TCP/IP and OSI thus differed in terms of technical details; but they also differed in terms of their context and legitimacy, one being a product of international-standards bodies, the other of military-funded research experiments. The technical and organizational differences imply different processes for standardization, and it is the peculiar nature of the so-called Requests for Comments ("RFC") process that gave TCP/IP one of its most distinctive features. The RFC system is widely recognized as a unique and serendipitous outcome of the research process of Arpanet.[56] In a thirty-year retrospective (published, naturally, as an RFC: RFC 2555), Vinton Cerf says, "Hiding in the history of the RFCs is the history of human institutions for achieving cooperative work."[57] He goes on to describe their evolution over the years:

> When the RFCs were first produced, they had an almost 19th century character to them—letters exchanged in public debating the merits of various design choices for protocols in

54. Clark, *supra* note 51, at 54.

55. *Id.* at 55.

56. RFCs are archived in many places, but the official site is RFC Editor, http://www.rfc-editor.org/ (last visited Feb. 27, 2009).

57. RFC Editor, 30 Years of RFCs 5 (Apr. 7, 1999), http://www.rfc-editor.org/rfc/rfc2555.txt.

> the ARPANET. As email and bulletin boards emerged from the fertile fabric of the network, the far-flung participants in this historic dialog began to make increasing use of the online medium to carry out the discussion—reducing the need for documenting the debate in the RFCs and, in some respects, leaving historians somewhat impoverished in the process. RFCs slowly became conclusions rather than debates.[58]

Increasingly, they also became part of a system of discussion and implementation in which participants created working software as part of an experiment in developing the standard, after which there was more discussion, then perhaps more implementation, and finally, a standard. The RFC process was a way to condense the process of standardization and validation into implementation; which is to say, the proof of open systems was in the successful connection of diverse networks, and the creation of a standard became a kind of ex post facto rubber-stamping of this demonstration. Any further improvement of the standard hinged on an improvement on the standard implementation because the standards that resulted were freely and widely available:

> A user could request an RFC by email from his host computer and have it automatically delivered to his mailbox. . . . RFCs were also shared freely with official standards bodies, manufacturers and vendors, other working groups, and universities. None of the RFCs were ever restricted or classified. This was no mean feat when you consider that they were being funded by DoD during the height of the Cold War.[59]

The OSI protocols were not nearly so freely available. The ironic reversal—the transparency of a military-research program versus the opacity of a Geneva-based international-standards organization—goes a long way toward explaining the reasons why geeks might find the story of TCP/IP's success to be so appealing. It is not that geeks are secretly militaristic, but that they delight in such surprising

58. *Id*. at 6.
59. *Id*. at 11.

reversals, especially when those reversals exemplify the kind of ad hoc, clever solution to problems of coordination that the RFC process does. The RFC process is not the only alternative to a consensus-oriented model of standardization pioneered in the international organizations of Geneva, but it is a specific response to a reorientation of power and knowledge that was perhaps more "intuitively obvious" to the creators of Arpanet and the Internet, with its unusual design goals and context, than it would have been to the purveyors of telecommunications systems with over a hundred years of experience in connecting people in very specific and established ways.

VII. SUCCESS AS FAILURE

By 1985, OSI was an official standard, one with widespread acceptance by engineers, by the government and military (the "GOSIP" standard), and by a number of manufacturers, the most significant of which was General Motors, with its Manufacturing Automation Protocol ("MAP"). In textbooks and handbooks of the late 1980s and early 1990s, OSI was routinely referred to as the inevitable standard—which is to say, it had widespread legitimacy as the standard that everyone should be implementing—but few implementations existed. Many of the textbooks on networking from the late 1980s, especially those slanted toward a theoretical introduction, give elaborate detail of the OSI reference model—a generation of students in networking was no doubt trained to understand the world in terms of OSI—but the ambivalence continued. Indeed, the most enduring legacy of the creation of the OSI protocols is not the protocols themselves (some of which, like ASN.1, are still widely used today), but the pedagogical model: the "7 layer stack" that is as ubiquitous in networking classes and textbooks as UNIX is in operating-systems classes.[60]

60. This can be clearly seen, for instance, by comparing the various editions of the main computer-networking textbooks. *See, e.g.*, DOUGLAS E. COMER, INTERNETWORKING WITH TCP/IP (five editions between 1991 and 2005); WILLIAM STALLINGS, DATA AND COMPUTER COMMUNICATIONS (eight editions between 1985 and 2006); ALFRED S. TANENBAUM, COMPUTER NETWORKS (four editions between 1981 and 2003).

But in the late 1980s, the ambivalence turned to confusion. With OSI widely recognized as the standard, TCP/IP began to show up in more and more actually existing systems. For example, in *Computer Network Architectures and Protocols*, Carl Sunshine says, "Now in the late 1980s, much of the battling seems over. CCITT and ISO have aligned their efforts, and the research community seems largely to have resigned itself to OSI."[61] But immediately afterward he adds:

> It is ironic that while a consensus has developed that OSI is indeed inevitable, the TCP/IP protocol suite has achieved widespread deployment, and now serves as a de facto interoperability standard. . . . It appears that the vendors were unable to bring OSI products to market quickly enough to satisfy the demand for interoperable systems, and TCP/IP were there to fill the need.[62]

The more implementations that appeared, the less secure the legitimate standard seemed to be. By many accounts the OSI specifications were difficult to implement, and the yearly networking-industry "Interop" conferences became a regular locale for the religious war between TCP/IP and OSI. The success of TCP/IP over OSI reflects the reorientation of knowledge and power to which Free Software is also a response. The reasons for the success are no doubt complex, but the significance of the success of TCP/IP illustrates three issues: availability, modifiability, and serendipity.

A. Availability

The TCP/IP standards themselves were free to anyone and available over TCP/IP networks, exemplifying one of the aspects of a recursive public: that the only test of participation in a TCP/IP-based internetwork is the fact that one possesses or has created a device that implements TCP/IP. Access to the network is contingent on the interoperability of the networks. The standards were not "published"

61. Carl A. Sunshine, *A Brief History of Computer Networking*, *in* COMPUTER NETWORK ARCHITECTURES AND PROTOCOLS 3, 5 (Carl A. Sunshine ed., 2d ed. 1989).

62. *Id.*

in a conventional sense, but made available through the network itself, without any explicit intellectual property restrictions, and without any fees or restrictions on who could access them. By contrast, ISO standards are generally not circulated freely, but sold for relatively high prices, as a source of revenue, and under the general theory that only legitimate corporations or government agencies would need access to them.

Related to the availability of the standards is the fact that the standards process that governed TCP/IP was itself open to anyone, whether corporate, military or academic. The structure of governance of the Internet Engineering Task Force ("IETF") and the Internet Society ("ISOC") allowed for anyone with the means available to attend the "working group" meetings that would decide on the standards that would be approved. Certainly this does not mean that the engineers and defense contractors responsible actively sought out corporate stakeholders or imagined the system to be "public" in any dramatic fashion; however, compared to the system in place at most standards bodies (in which members are usually required to be the representatives of corporations or governments), the IETF allowed individuals to participate qua individuals.[63]

B. Modifiability

Implementations of TCP/IP were widely available, bootstrapped from machine to machine along with the UNIX operating system and other tools (e.g., the implementation of TCP/IP in BSD 4.2, the BSD version of UNIX), generally including the source code. An existing implementation is a much more expressive and usable object than a specification for an implementation, and though ISO generally prepares reference implementations for such standards, in the case of OSI there were many fewer implementations to work with or build on. Because multiple implementations of TCP/IP already existed, it was easy to validate: did your (modified) implementation work with the other existing implementations? By contrast, OSI would provide

63. The structure of the IETF, the Internet Architecture Board, and the ISOC is detailed in COMER (2d ed. 1991), *supra* note 60, at 9–11; *see also* SCHMIDT & WERLE, *supra* note 42, at 53–56.

independent validation, but the in situ validation through connection to other OSI networks was much harder to achieve, there being too few of them, or access being restricted. It is far easier to build on an existing implementation and to improve on it piecemeal, or even to rewrite it completely, using its faults as a template (so to speak), than it is to create an implementation based solely on a standard. The existence of the TCP/IP protocols in BSD 4.2 not only meant that people who installed that operating system could connect to the Internet easily, at a time when it was by no means standard to be able to do so, but it also meant that manufacturers or tinkerers could examine the implementation in BSD 4.2 as the basis for a modified, or entirely new, implementation.

C. Serendipity

Perhaps most significant, the appearance of widespread and popular applications that were dependent on TCP/IP gave those protocols an inertia that OSI, with relatively few such applications, did not have. The most important of these by far was the World Wide Web (the http protocol, the HTML mark-up language, and implementations of both servers, such as libwww, and clients, such as Mosaic and Netscape). The basic components of the Web were made to work on top of the TCP/IP networks, like other services that had already been designed (ftp, telnet, gopher, archie, etc.); thus, Tim Berners-Lee, who co-invented the World Wide Web, could also rely on the availability and openness of previous work for his own protocols. In addition, Berners-Lee and CERN (the European Organization for Nuclear Research) dedicated their work to the public domain more or less immediately, essentially allowing anyone to do anything they wished with the system they had cobbled together.[64] From the perspective of the tension between TCP/IP and OSI, the World Wide Web was thus what engineers call a "killer app," because its existence actually drove individuals and corporations to make decisions (in favor of TCP/IP) that it might not have made otherwise.

64. *See* TIM BERNERS-LEE, WEAVING THE WEB: THE ORIGINAL DESIGN AND ULTIMATE DESTINY OF THE WORLD WIDE WEB (1999) (detailing the origins of the World Wide Web).

CONCLUSION

Openness and open systems are key to understanding the practices of Free Software: the open-systems battles of the 1980s set the context for Free Software, leaving in their wake a partially articulated infrastructure of operating systems, networks, and markets that resulted from figuring out open systems. The failure to create a standard UNIX operating system opened the door for Microsoft Windows NT, but it also set the stage for the emergence of the Linux-operating-system kernel to emerge and spread. The success of the TCP/IP protocols forced multiple competing networking schemes into a single standard—and a singular entity, the Internet—that carried with it a set of built-in goals that mirror the moral-technical order of Free Software.

This "infrastructure" is at once technical (protocols and standards and implementations) and moral (expressing ideas about the proper order and organization of commercial efforts to provide high-tech software, networks, and computing power). As with the invention of UNIX, the opposition commercial-noncommercial (or its doppelgangers public-private, profit-nonprofit, capitalist-socialist, etc.) doesn't capture the context. Constraints on the ability to collaborate, compete, or withdraw are in the making here through the technical and moral imaginations of the actors involved: from the corporate behemoths like IBM to (onetime) startups like Sun to the independent academics and amateurs and geeks with stakes in the new high-tech world of networks and software.

The creation of a UNIX market failed. The creation of a legitimate international networking standard failed. But they were local failures only. They opened the doors to new forms of commercial practice (exemplified by Netscape and the dotcom boom) and new kinds of politicotechnical fractiousness (ICANN, IPv6, and "net neutrality"). But the blind spot of open systems-intellectual property—at the heart of these failures also provided the impetus for some geeks, entrepreneurs, and lawyers to start figuring out the legal and economic aspects of Free Software, and it initiated a vibrant experimentation with copyright licensing and with forms of innovative coordination and collaboration built on top of the rapidly spreading protocols of the Internet.